MP24 @ II UWr 22 lutego 2024 r.

Lista zadań nr 1

Zadanie 1.

Przeanalizuj poniższą sekwencję wyrażeń. Jaki wynik wypisze interpreter w odpowiedzi na każde z nich, zakładając, że będą obliczane w kolejności w której są podane? Sprawdź swoje przewidywania używając interpretera.

```
10;;

2 * 3 + 4;;

2 * (3 + 4);;

let a = 3;;

let b = a + 1;;

if b > a && b < a * b then b else a;;

2 + (if b > a then b else a);;

let a = if b * b > b then b else -1 in a * b;;
```

Zadanie 2. (3 pkt)

Dla każdego z poniższych wyrażeń stwierdź, czy jest poprawnie otypowane, jeśli tak, to podaj jego typ. Uzasadnij swoją odpowiedź.

```
"foo" ^ 42;

"foo" ^ string_of_int 42;;

1. = 2;;

fun a -> a + 5;;
```

```
fun a -> if a > 5 then a else "foo";;

fun a b -> if a > 5 then a else b;;

fun a b ->
    let c = a = b in
    if a > 3 && b = "foo"
    then c
    else false;;

fun a ->
    let f a b = a * a + b * b in
    f (f a a) a;;

let f a = a > 2 in
if 3 > 2 then true else f (f 2);;
```

Zadanie 3. (3 pkt)

W poniższych wyrażeniach zlokalizuj wolne i związane wystąpienia zmiennych. Które wystąpienia wiążą każde z wystąpień związanych?

```
x;;

let x = 3 in x + y;;

let x = 1
and y = x + 2
in x + y;;

let x = 1 in
let y = x + 2 in
x + y;;

let f x y = x * y * z;;

let f x =
    let g y z = x * y * z in
g (h x) z;;
```

Zadanie 4. (2 pkt)

Zdefniuj funkcję o trzech argumentach będących liczbami całkowitymi, której wynikiem jest suma kwadratów dwóch większych jej argumentów.

Zadanie 5.

Zauważ że w naszym modelu obliczania wartości dopuszczamy, aby operatorami były wyrażenia złożone. Korzystając z tej obserwacji, wyjaśnij działanie następującej funkcji:

```
let a_plus_abs_b a b =
    (if b > 0 then (+) else (-)) a b
```

Zadanie 6. (2 pkt)

Przeanalizuj poniższe funkcje. W jaki sposób możesz użyć ich, aby sprawdzić, czy interpreter wykonuje obliczenia używając gorliwej, czy leniwej kolejności obliczania? Uzasadnij odpowiedź pokazując, jak interpreter wyliczyłby wartość w zależności od kolejności obliczania. Załóż, że reguła obliczania wartości wyrażenia `if` nie zależy od kolejności obliczania.

```
let rec f () = f ()

let test x y =
  if x = 0 then 0 else y
```